

## TITLE OF THE INVENTION

### CHECKSUM GENERATION APPARATUS AND METHOD THEREOF

## CROSS-REFERENCE TO RELATED APPLICATIONS

**[0001]** This application claims the benefit of PCT International Patent Application No. PCT/KR2004/003249, filed December 10, 2004, and Korean Patent Application No. 2003-91882, filed December 16, 2003, in the Korean Intellectual Property Office, the disclosures of which are incorporated herein by reference.

## BACKGROUND OF THE INVENTION

### 1. Field of the Invention

**[0002]** Aspects of the present invention relate to data processing, and more particularly, to a checksum generation apparatus and a method thereof used to determine whether data is transmitted and received without error.

### 2. Description of the Related Art

**[0003]** A checksum calculation counts a bit number in a transmission unit of data in order to determine whether received data has a same bit number as data transmitted by a transmitter. If a checksum calculated by a receiver matches a checksum transmitted by the transmitter, the data is determined to have been received without error. Checksum calculation and determination are performed in a transmission control protocol (TCP) and a user datagram protocol (UDP) of internet protocols.

**[0004]** The checksum calculation is an important data processing operation in the internet and the checksum must be calculated at a high speed. A conventional checksum calculation method has a problem of a low calculation speed, since data is processed in units of 16 bits.

## SUMMARY OF THE INVENTION

**[0005]** Aspects of the present invention provide a checksum generation apparatus and a method thereof which improves a checksum calculation speed by performing an addition in units of 32 bits or more and converting an addition result to a 16-bit checksum.

**[0006]** According to an aspect of the present invention, a checksum calculation speed is increased by not dividing input data. In particular, when the checksum generation apparatus is implemented in an ASIC device, a calculation speed of a checksum generation apparatus is increased by using a 32-bit or a 64-bit adder in a library of the ASIC device.

**[0007]** According to an aspect of the present invention, a checksum generation apparatus includes a control unit which, in response to information on a predetermined length, outputs a control signal when an amount of data corresponding to the predetermined length is received; an addition unit which performs an addition on the received data and in response to the control signal, outputs an addition result; and a conversion unit which converts the addition result to a checksum.

**[0008]** The addition unit may receive data in units of 32 bits or more (integer multiples of 16 bits) and perform the addition on the received data.

**[0009]** The conversion unit may divide the addition result into a sum and a carry, partition the sum into 16-bit segments, and add the 16-bit segments to the carry, thereby obtaining a final sum.

**[0010]** The conversion unit may include a partial sum addition unit excluding a carry from the addition result, partitioning the carry-excluded addition result into 16-bit segments, and adding the 16-bit segments, thereby obtaining a partial sum; a first adder adding the carry to the partial sum to obtain a second addition result; a second adder adding the second addition result and a second carry occurring in the second addition result to obtain a third addition result; and a complement calculator outputting a 1's complement value of the third addition result.

**[0011]** According to another aspect of the present invention, a method of generating a checksum includes adding input data until a predetermined control signal is received; outputting a sum and a carry obtained from the addition result when the control signal is received; and adding the sum and the carry to obtain a second addition result and converting the second addition result to a checksum.

**[0012]** In the adding of the data, the data may be received in units of 32 bits or more (integer multiples of 16 bits) and an addition be performed on the received data.

**[0013]** The outputting of the sum and the carry may include adding the received data in units of 32 bits or more (integer multiples of 16 bits); and adding carries generated in the adding of the received data in the units of 32 bits or more.

**[0014]** Further, the adding of the sum and the carry and the converting of the second addition result to the checksum may further include excluding a first carry from the first addition result, partitioning the carry-excluded first addition result into 16-bit segments, adding the 16-bit segments, thereby obtaining a partial sum; adding the first carry to the partial sum to obtain a second addition result and a second carry; adding the second addition result and the second carry to obtain a third addition result; and outputting a 1's complement value of the third addition result.

**[0015]** According to still another aspect of the present invention, a computer-readable storage medium comprises recording areas storing a program executable by a computer, the program comprising instructions for enabling a computer to add input data until a predetermined control signal is received; output a sum and a carry obtained from an addition result when the control signal is received; and add the sum and the carry and convert the added sum and carry to a checksum.

**[0016]** Additional aspects and/or advantages of the invention will be set forth in part in the description which follows and, in part, will be obvious from the description, or may be learned by practice of the invention.

#### BRIEF DESCRIPTION OF THE DRAWINGS

**[0017]** These and/or other aspects and advantages of the invention will become apparent and more readily appreciated from the following description of the embodiments, taken in conjunction with the accompanying drawings of which:

FIG. 1 is a view for explaining a method of generating a checksum;

FIG. 2 is a block diagram illustrating a checksum generation apparatus according to an embodiment of the present invention;

FIG. 3 is a detailed block diagram illustrating a checksum generation apparatus using a 32-bit adder;

FIG. 4 is a detailed block diagram illustrating a checksum generation apparatus using a

64-bit adder;

FIG. 5A is a view illustrating a TCP segment format;

FIG. 5B is a view illustrating a pseudo header format; and

FIG. 6 is a flowchart illustrating a method of generating a checksum according to an embodiment of the present invention.

#### DETAILED DESCRIPTION OF THE EMBODIMENTS

**[0018]** Reference will now be made in detail to the present embodiments of the present invention, examples of which are illustrated in the accompanying drawings, wherein like reference numerals refer to the like elements throughout. The embodiments are described below in order to explain the present invention by referring to the figures.

**[0019]** FIG. 1 is a view for explaining a method of generating a checksum according to an embodiment of the present invention.

**[0020]** In the method as illustrated in FIG. 1, a 16-bit adder is used for generating the checksum. As shown in FIG. 1, data (0001 f203 f4f5 f6f7) is divided into 16-bit data segments and a first addition is performed on the 16-bit data segments. That is, the first addition is 0001 + f203 + f4f5 + f6f7. The first addition result, Sum1, is 2ddf0. Since the 16-bit adder is used, a carry of the first addition value Sum 1 is 2. In a second addition, the carry of the first addition result Sum1 is added back to the 16-bit adder to obtain a second addition result, Sum2, that is, a final value: 2 + ddf0 = ddf2. The checksum is obtained by performing a 1's complement operation on the final value ddf2. Therefore, the checksum is 220d.

**[0021]** FIG. 2 is a block diagram illustrating a checksum generation apparatus according to an embodiment of the present invention.

**[0022]** The checksum generation apparatus shown in FIG. 2 comprises an addition unit 210, a control unit 220, and a conversion unit 230. Data in units of a 32-bit or 64-bit segment is input to the addition unit 210. In some cases, the data may be input in units of more than 64-bit segment. The addition unit 210 may be constructed with a 32-bit or 64-bit adder. Alternatively, the addition unit 210 may be constructed with an 80-bit, 96-bit, or 128-bit adder. In the embodiment shown in FIG. 2, the input data is not divided into 16-bit data segments but added

as it is received. In response to information on a predetermined input data length, the control unit 220 determines whether the input data corresponding to the input data length is received and accumulated. When the input data corresponding to the length is received, the control unit 220 outputs a control signal to the addition unit 210. In response to the control signal, the addition unit 210 outputs an addition value to the conversion unit 230. The conversion unit 230 converts the addition value to a 16-bit checksum and outputs the checksum.

**[0023]** The addition unit 210 accumulates and adds the input data to obtain a partial sum until the control signal is received from the control unit 220. A carry addition is separately performed. In consideration of a maximum value of the carry, a result of the carry addition, that is, a carry sum, is stored in units of 10 bits. When the control signal is received from the control unit 220, the partial sum and the carry sum are output to the conversion unit 230. The conversion unit 230 divides the partial sum in 16-bit segments and adds the 16-bit segments. A result of the addition is added back to the carry sum to obtain the final sum. The conversion unit 230 outputs a 1's complement value of the final sum. For example, assuming that the addition unit 210 performs the addition in units of 32 bits and the partial sum is a 32-bit value, the conversion unit 230 obtains the final sum by adding higher 16 bits and lower 16 bits of the 32-bit partial sum to the carry. In addition, assuming that the addition unit 210 performs the addition in units of 64 bits and the partial sum is a 64-bit value, the conversion unit 230 obtains the final sum by adding most significant 16 bits, higher 16 bits, lower 16 bits, and least significant 16 bits of the 64-bit partial sum to the carry.

**[0024]** FIG. 3 is a detailed block diagram illustrating an example of the checksum generation apparatus shown in FIG. 2 using a 32-bit adder.

**[0025]** As shown in FIG. 3, the addition unit 210 comprises a 32-bit adder 305 and a carry adder 310. The 32-bit adder 305 adds input data, and the carry adder 310 adds carries. The conversion unit 230 comprises a partition adder 315, a first adder 320, a second adder 325, and a complement calculator 330. The partition adder 315 partitions 32-bit data into a high 16-bit data segment and a low 16-bit data segment and adds the 16-bit data segments. The addition result is added back to a carry output from the carry adder 310 in the first adder 320. The second adder 325 adds an addition result of the first adder 320 to a carry of the first adder 320 and outputs a final sum, which has a 16 bit value. The complement calculator 330 outputs a 1's

complement value of the final sum as the checksum.

**[0026]** FIG. 4 is a detailed block diagram illustrating an example of the checksum generation apparatus shown in FIG. 2 using a 64-bit adder.

**[0027]** As shown in FIG. 4, the addition unit 210 comprises a 64-bit adder 405 and a carry adder 410. The 32-bit adder 405 adds input data, and the carry adder 410 adds carries of the adder 405. The conversion unit 230 comprises a partition adder 415, a first adder 420, and a second adder 425, and a complement calculator 430. The partition adder 415 partitions 62-bit data into four 16-bit data segments and adds the four 16-bit data segments. The addition result is added back to a carry output from the carry adder 410 in the first adder 420. The second adder 425 adds an addition result of the first adder 420 to a carry of the first adder 420 and outputs a final sum, which has a 16 bit value. The complement calculator 430 outputs a 1's complement value of the final sum as the checksum.

**[0028]** The checksum calculation described above may be employed to any protocols such as Internet Protocol (IP), Transmission Control Protocol (TCP), and User Datagram Protocol (UDP). Now, a checksum calculation employed in the protocol TCP will be described in detail with reference to FIGS. 5A and 5B.

**[0029]** FIG. 5A is a view illustrating a TCP segment format.

**[0030]** The TCP segment includes a TCP header 510 and a TCP payload 520. The TCP header 510 includes a checksum field where a checksum is entered. A pseudo header is needed in order to calculate the checksum in a protocol TCP. Other fields in the TCP segment are well known to persons skilled in the art and detailed descriptions thereof are not included.

**[0031]** FIG. 5B is a view illustrating a pseudo header format.

**[0032]** A pseudo header comprises a source IP address, a destination IP address, padding, a protocol number, and a TCP packet length. The pseudo header is not actually transmitted but is used to calculate a checksum of a TCP packet. An end portion of data has a value of 0 by a padding operation in order that the data length is a multiple of 16 bits. A checksum field of the TCP header has a value of 1. An addition is performed in units of 16 bits. A 1's complement of the addition result is entered into the checksum field. When receiving TCP segments, the

receiver obtains an IP address from an IP header, produces a TCP pseudo header, and calculates a checksum.

**[0033]** FIG. 6 is a flowchart illustrating a method of generating a checksum according to an embodiment of the present invention.

**[0034]** The input data is added in units according to a character of the input data in operation S610. That is, for example, if the input data is 32 bit input data, the data is added in 32 bit units. Carries occurring in addition results are separately added together. The addition result of the carries is stored in units of 10 bits in consideration of a maximum value of the carries. It is determined whether the input data corresponding to a predetermined input data length is received and accumulated in operation S620. The input data is added until the input data corresponding to the predetermined length is received. The addition result is converted to a 16-bit value in operation S630. A 1's complement of the 16-bit value is output at operation S640.

**[0035]** Now, the operation S630 in a case where the input data has a 32-bit value will be described in detail. The addition result and a carry of the addition result are indicated by Sum1 and Carry1, respectively. Therefore,  $\text{Temp1} = (\text{16-bit MSB of Sum1}) + (\text{16-bit LSB of Sum1}) + \text{Carry1}$ . If a carry of Temp1 is indicated by Carry2,  $\text{Temp2} = (\text{Temp1}) + (\text{Carry2})$ . The value Temp1 becomes a 16-bit final value. In a case where the input data has a 64-bit value, the Sum1 and the Carry1 are obtained by adding 64 bit values and the value Temp1 is obtained by adding four 16-bit segments of the Sum1 and the Carry1. The value Temp2 is obtained by the calculation  $\text{Temp2} = (\text{Temp1}) + (\text{Carry2})$ , as described above.

**[0036]** The above described method of generating a checksum may be implemented with a program which is executable on a computer. Codes and code segments constituting the program may be written by a person skilled in the art based on the disclosure herein. In addition, the program may be stored in a computer-readable storage medium and read and executed by a computer. The computer-readable storage medium may include a magnetic storage medium, an optical storage medium, and a carrier wave medium.

**[0037]** Although a few embodiments of the present invention have been shown and described, it would be appreciated by those skilled in the art that changes may be made in this embodiment without departing from the principles and spirit of the invention, the scope of which is defined in the claims and their equivalents.